Simple Data-Driven Context-Sensitive Lemmatization *

Grzegorz Chrupała
National Centre for Language Technology
Dublin City University
Glasnevin, Dublin 9, Ireland
grzegorz.chrupala@computing.dcu.ie

Resumen: Para los idiomas con una morfología flexiva rica, la lematizacíon es uno de los pasos básicos e indispensables para su tratamiento automático. En este artículo presentamos un método de lematizaciíon sencillo basado en el aprendizaje automático y que tiene en cuenta el contexto de las formas en el texto. Tratamos la lematización como una tarea de clasificación e inducimos las etiquetas de clases de forma automática. Para este fin calculamos el script de ediciíon más corto (SES) entre las cadenas invertidas de entrada y de salida. Un SES describe las modificaciones que se deben aplicar a la cadena de entrada (la forma) para convertirla en la cadena de salida (el lema). Con nuestro método conseguimos unos resultados competitivos en una serie de lenguas tipologicamente diversas.
Palabras clave: lematización, aprendizaje automático

Abstract: Lemmatization for languages with rich inflectional morphology is one of the basic, indispensable steps in a language processing pipeline. In this paper we present a simple data-driven context-sensitive approach to lemmatizating word forms in running text. We treat lemmatization as a classification task for Machine Learning, and automatically induce class labels. We achieve this by computing a Shortest Edit Script (SES) between reversed input and output strings. A SES describes the transformations that have to be applied to the input string (word form) in order to convert it to the output string (lemma). Our approach shows competitive performance on a range of typologically different languages.
Keywords: lemmatization, machine learning

1 Introduction

Lemmatization and morphological analysis are traditionally performed using dictionary lookup and/or rule-based finite-state methodology. Data-driven approaches to syntactic analysis (parsing) are a very active area of research: in comparison relatively little has been done to apply similar methodology to morphology. One of the reasons for this state of affairs may be the fact that the language on which most research is published, i.e. English, does not have a complex inflectional morphology, and lemmatization and inflectional analysis can be done using relatively simple-minded approaches. In this paper we will present a data-driven, corpus-based approach to lemmatization that has several advantages over alternative lexicon or rule based methods. It requires less effort on the part of human experts and is to a large extent language-independent. In contrast to methods which rely on dictionary-lookup, it robustly generalizes to unseen word forms.

* We gratefully acknowledge support from Science Foundation Ireland grant 04/IN/I527 for the research reported in this paper.

1.1 Related work

Van den Bosch and Daelemans (1999) describe using memory-based classification to perform morphological analysis of Dutch words. They aim to cover both inflectional and derivational morphology, and to obtain full analyses of word forms. Their training data is derived from a lexical database (CELEX) rather than a corpus of running text, which means that their model does not take surrounding context into account. They cast the task as classification by assigning class labels to individual characters. The induction of those classes seems to require human intervention. This work is not directly comparable to the one presented in this paper, as van den Bosch and Daelemans have a more ambitious goal, not

limited to lemmatization, and they use a lexicon rather than a text corpus for both training and evaluation.

A different take on using machine-learning to perform morphological analysis is to avoid casting it as a classification task. Stroppa and Yvon (2005) use the paradigm of analogical learning, which permits the model to produce structured outputs rather than just class labels. They are thus able to map input strings (word forms) to trees representing their morphological structure. Their work is also not directly comparable to our results for the same reasons as those mentioned above.

Erjavec and Džeroski (2004) present an approach to lemmatization for Slovene. They decompose the task into two parts – they first perform morphosyntactic tagging on input text and then provide the lemmatizer with the word form plus the morphosyntactic tag. They use a trigram tagger for the first subtask and a first-order decision list learning system for the second. The lemmatizer is trained on a lexicon containing 15,000 lemmas with their full inflectional paradigm. They report an accuracy of 92% on unknown inflecting Slovene words.

2 Lemmatization as a classification task

Many popular machine-learning methods (i.e. memory-based learning or large-margin classifiers) require that the process to be performed be cast as a classification task. The training data should consist of a collection of examples (represented as vectors of features) with assigned class labels. The algorithms learn to assign those labels to new examples. Even though classification is not the only available paradigm for machine-learning (cf Stroppa and Yvon (2005)), nevertheless some of the most performant methods such as Support Vector Machines (Boser, Guyon, and Vapnik, 1992), do assume it. In this paper we show how lemmatization can be easily adapted to the classification setting, given some reasonable assumptions about the data.

It is not immediately obvious what the class labels should be for the task of lemmatization. In previous approaches to morphological analysis and lemmatization classes are specified manually, on the basis of analysis of inflectional or derivational paradigms for a given language. This works but is labor-intensive.

The approach we propose is to derive the classes automatically from training data. Instead of inspecting data to identify and specify paradigms we try to discover recurring patterns in the mappings form word forms to lemmas. We present a very simple class-inference mechanism based on the idea of the *Shortest Edit Script* (SES) between two strings (Myers, 1986). Finding a SES between two sequences is a dual problem to finding the *longest common subsequence* (Aho, Hirschberg, and Ullman, 1976; Hirschberg, 1977). An edit script of sequences a and b is a set of *instructions* (insertions and deletions) which, when applied to sequence a, transform it into sequence b. An instruction specifies whether an insertion or a deletion should be performed, at which position in sequence a, and which element is to be inserted or deleted. As an example consider the strings $a = pidieron$ and $b = pedir$[1]. A SES that transforms a into b is $\{\langle \mathrm{D}, i, 2\rangle, \langle \mathrm{I}, e, 3\rangle, \langle \mathrm{D}, e, 5\rangle, \langle \mathrm{D}, o, 7\rangle, \langle \mathrm{D}, n, 8\rangle\}$. This is interpreted as

- delete character i at position 2
- insert character e before position 3
- delete character e at position 5
- delete character o at position 7
- delete character n at position 8

We use the SES between word forms and their lemmas as class labels.

One nuance is that in the majority of languages inflectional morphology is mostly suffixal, i.e. it affects the endings of words, or occasionally material in word roots, rather than the beginning [2]. This means that SESs will work better as classes if we index characters starting at the end of the string rather than at the beginning, or equivalently if we compute the SES on reversed strings. For example if we compute $ses(repitieron, repetir)$ [3] we will not get the same SES as for the example above (as all indices will be incremented by 2). However on reversed strings, $ses(noreidip, ridep)$ and $ses(noreitiper, riteper)$ give the same result $\{\langle \mathrm{D}, n, 1\rangle, \langle \mathrm{D}, o, 2\rangle, \langle \mathrm{D}, e, 4\rangle, \langle \mathrm{D}, i, 7\rangle, \langle \mathrm{I}, e, 8\rangle\}$. This accords with the linguistic notion that the strings *pedir* and *repetir* are forms of Spanish verbs which occupy the same position in the verb inflection paradigm of the same conjugation class. If our assumptions about inflectional morphology hold, i.e. if it is predominantly suffixal, such agreement should happen frequently.

[1] *pidieron* is the 3rd person plural preterite form of the verb *pedir*, ASK in Spanish.

[2] One well-known exception is Irish.

[3] *repitieron* is the 3rd person plural preterite form of the verb *repetir*, REPEAT in Spanish.

3 Experiments

We have performed a series of experiments on a range of languages and data sets to evaluate how our idea works in practice.

3.1 Data

We have trained the classifier on lemma-annotated corpora in eight languages.

- Spanish, Cast3LB (Civit and Martí, 2004)
- Catalan, Cat3LB (Civit, Bufí, and Valverde, 2004)
- Portuguese, Bosque 7.3, (Afonso et al., 2002)
- French, Paris-7 Treebank, (Abeillé, Clément, and Toussenel, 2003)
- Polish, Polish Frequency Corpus, Section B - Press, (Bień and Woliński, 2003)
- Dutch, Alpino Treebank, (van der Beek et al., 2002)
- German, Tiger Treebank (Brants et al., 2002)
- Japanese, Kyoto Text Corpus (Kurohashi and Nagao, 2003)

For each corpus we took 10,000 tokens as the test test, another 10,000 tokens as development set, and 70,000 as training set.

In the Japanese corpus the word forms appear in Kanji and the lemmas in Hiragana. Since our method needs data written in the same script, and preferably an alphabetic one, we convert both Kanji and Hiragana to Romanji using the Kakasi software package[4]. We have not evaluated the accuracy of the conversion so there may be some noise in our Japanese results.

3.2 Methodology

For each language we use the same set of features. They are the following:

- The last 12 characters of the word form (i.e. 12 separate features)
- The target token word form (treated atomically)
- Word form of preceding 3 tokens and following 3 tokens

[4]Available for download at http://kakasi.namazu.org/. We would like to thank Masanori Oya for poiting out Kakasi to us and for help with the conversion.

The feature set was selected based on experimentation with the Spanish development set.

In our experiments we use one of the better-performing classifying machine-learning algorithms, i.e. Support Vector Machines (Boser, Guyon, and Vapnik, 1992; Vapnik, 1998). We use the LIBSVM implementation of Chang and Lin (2001). It implements the one-against-one strategy for non-binary (multi-class) classification. We binarize the features described above for use with SVM: i.e. each original feature-value combination is mapped to a new binary feature.

We used the default kernel (RBF). The parameters C (32768) and γ (3.05×10^{-5}) were chosen by cross-validation on the Spanish development set. Because we didn't repeat feature selection and parameter tuning separately for each language, our results may underestimate the potential performance of our method for languages other than Spanish.

When calculating the SES for word form - lemma pairs, we lowercase both strings and remove embedded quotes (occasionally found in German compound words). These simplifications reduce the number of classes and make the learning task easier. We also ignore lemma capitalization for evaluation.

4 Evaluation results

Table 1 presents the results of evaluation for all the languages on the test sets. The most straightforward performance metric is token accuracy: i.e. what proportion of tokens were correctly lemmatized (shown in the second column). The problem with this metric is that, depending on the data set, high accuracy can be achieved by simply returning the word form (the baseline method). E.g. for Japanese, where the only open-class words which inflect are verbs, the baseline method give 88.42% accuracy on the test set. Baseline accuracies are shown in the first column.

To give a more informative indication of the performance we also calculate precision, recall, and the harmonic mean of those two, the f-score. For those metrics we consider the empty SES, i.e. when lemma is equal to the word form, as the null class. The number of correct lemmas, excluding the nulls, are the true positives. Recall is then calculated by dividing the number of true positives by the number of non-null lemmas in gold standard, whereas precision is the number of true positives divided by the number of non-nulls among the predicted lemmas.

	Base Acc.	Accuracy	Precision	Recall	F-score
Catalan	66.33	97.27	95.91	93.40	94.64
German	52.64	95.11	94.61	92.03	93.31
Polish	48.61	95.06	93.75	91.96	92.84
Spanish	69.50	96.44	92.32	92.65	92.48
Japanese	88.42	98.36	94.05	88.77	91.33
Portuguese	71.74	96.38	91.85	90.58	91.21
French	61.88	94.36	92.16	87.93	89.99
Dutch	78.80	94.15	85.38	79.62	82.40

Table 1: Lemmatization evaluation for eight languages

	Base Acc.	Accuracy	Precision	Recall	F-score
Polish	26.31	80.29	81.73	77.53	79.58
Spanish	58.75	86.35	77.97	79.39	78.67
Portuguese	58.58	85.17	76.35	70.32	73.21
Catalan	60.16	82.99	76.11	66.05	70.72
German	55.95	78.88	72.02	62.80	67.09
Japanese	78.96	89.54	74.62	59.88	66.44
French	55.80	76.83	71.42	55.71	62.60
Dutch	66.42	72.40	46.82	31.33	37.54

Table 2: Lemmatization evaluation for eight languages – unseen word forms only

Except for one case, the f-scores cluster between 90% and 95%, even though base accuracies range from under 50% to almost 90%. Thus even though the languages represent varying degrees of inflectional richness, this has limited impact on the performance of the SES-based lemmatization method.

There is one outlier, however: for Dutch the f-score is over 7% worse than the next worst result. We suspect that this is due to the fact that for this dataset our assumption of predominatly suffixal inflection does not hold. It turns out that there are many tokens in the Dutch corpus where mapping the word form to lemma involves changes to the beginning of the string, often involving moving an initial part of the word form to the end. This happens in the case of verbs with separable particles such as: *lesgegeven* → *geef_les* or *meelopen* →*loop_mee*. In those two examples the lemma is the verb inflected for present first-person singular, with the separable particle following it. Even more problematic are cases where the separable particle that appears at another point in the utterance is appended at the end of the lemma, e.g. the sequence of word forms *we trokken erop uit* is lemmatized as follows: $\langle we, trek\_uit, erop, uit \rangle$. Another non-final transformation involves compound words, where compounding morpheme *-s* is replaced by an underscore in the lemma: *verbrandingsmotor* → *verbranding_motor*. It is clear that for such transformations classifying examples by SES on reversed strings is not sufficient.

It is also evident, however, that for many datasets such cases are rare and our simple method shows reasonable performance. The problems with lemmatizing the Dutch corpus are only partly caused by the features of the language itself. Equally important are the choices made by corpus designers. This can be seen by comparing our results for Dutch to those on the closely related language German, which also has verbs with separable particles and the compounding morpheme *-s-*. However in the German Tiger treebank separable particles which appear elsewhere in the text are not attached to the verb lemma, and the morpheme *-s-* does not get replaced by an underscore in lemmas for compound nouns. For example the sentence *Konzernchefs lehnen den Milliardär als US-Präsidenten ab* is lemmatized as ⟨*Konzernchef, lehnen, der, Milliardär, als, US-Präsident, ab*⟩ even though it contains the verb *ablehnen* with the separable particle *ab*. It could be plausibly argued that in the common "pipeline" approach to language processing finding such non-local dependencies is best left to the syntactic level of analysis.

Table 2 shows the same statistics as Table 1 for the subset of word forms not seen in the training set. There is more variance in these results than for the all-tokens evaluation and the relative ranking of languages is also different.

Understandably, for all test-sets there is a significant drop in the f-score for the unseen subset. There is a group of languages where the difference is between the f-scores is below 20% (Polish, Spanish and Portuguese), another group (Catalan, German, Japanese and French) between 20% and 30%, and again the oulier datapoint of Dutch, where the difference is of 44%. It remains to be investigated to what degree these differences are a function of the morphological features of the languages in question and to what extent they reflect the nature of the particular datasets or treenbanks used in this evaluation.

4.1 Comparison to Freeling

In order to determine how the SES-based machine-learning approach to lemmatization compares to more traditional methods, we compare the results of our system to the performance of a popular analyzer Freeling (Carreras et al., 2004). Freeling performs a range of language-processing tasks (tokenization, morphological analysis, named-entity recognition, chunking etc.) for several languages. Below we compare the two systems on the lemmatization task on the Spanish and Catalan test sets.[5] Lemmatization in Freeling is based on lexicon lookup combined with disambiguation based on the part of speech tag in cases where the same word form can correspond to different lemmas. The Spanish lexicon size is about 71,000 word forms. The Catalan lexicon contains around 46,000 word forms.

In the input to Freeling we keep the original tokenization and sentence splits present in the corpus data. In the Spanish and Catalan treebanks multi-word expressions, named-entities, dates and quantities are treated as single tokens – we also keep those tokens, and consequently deactivate named-entity, multi-word, date and quantity handling by Freeling.

Tables 3 (Spanish) and 4 (Catalan) show the results for Freeling and for the SES-based method using the SVM classifier. We report results on all tokens and also results on tokens not seen in our training set. For Spanish, our system outperforms Freeling by about 2% on all tokens. There is a much bigger difference between the systems – 6.33% – on the unseen subset of tokens. For Catalan the differences are larger: 6.37% on all tokens and 10.58% on the unseen token subset. The poorer performance of Freeling on the Catalan data probably reflects its small lexicon size for that language.

[5]These results are for Freeling version 1.2. A new improved version has been recently released which might give different results.

	Different	SES better	p-value
Spanish	581	360	4.414×10^{-9}
Catalan	705	550	2.2×10^{-16}

Table 5: Statistical significance test

Freeling's lemmatization is not data-driven, and does not use the training data. The sharp drop in performance it shows for this subset is probably due to the fact that any tokens unseen in our training data are relatively uncommon words, in many cases probably absent from Freeling's word form lexicon. In those cases, Freeling simply returns the word form as the lemma, whereas our machine-learning model generalizes to unseen data and in most cases outputs the correct answer.

To determine how statistically significant the difference between the systems' performance is, we did a sign test on the results for the all tokens comparison. We adopt a confidence level of 99% ($\alpha = 0.01$) for these tests.

For each token we check whether the methods give different answers – for these cases (i.e. the number of trials) we calculate in how many cases the second method is better than the first (i.e. the number of successes). We then perform the binomial test with the null hypothesis that the probability of success is chance ($= 0.5$) and the alternative hypothesis that the probability of success is greater than chance (> 0.5). The results are summarized in Table 5. For both languages the p-values are much below α, so for our chosen confidence level the improvement is statistically significant.

5 Conclusions

The Shortest Edit Script approach to learning to lemmatize running text is appealingly simple and manages to combine good performace with a high degree of language independence. Other methods often rely on a large full-paradigm inflectional lexicon, either to perform word form lookup, or as a training resource. To train our system only a lemmatized corpus is needed – such resources are readily available for a number of languages. The simplicity and high language independence of our approach is evidenced by

		Base Acc.	Accuracy	Precision	Recall	F-score
Freeling	all	69.50	95.05	92.78	88.13	90.39
	unseen	58.75	82.05	82.58	64.36	72.34
SES+SVM	all	69.50	96.44	92.32	92.65	92.48
	unseen	58.75	86.35	77.97	79.39	78.67

Table 3: Comparison of SES+SVM to Freeling on the lemmatization task for Spanish

		Base Acc.	Accuracy	Precision	Recall	F-score
Freeling	all	66.33	93.32	93.08	83.93	88.27
	unseen	60.16	77.16	86.31	46.13	60.13
SES+SVM	all	66.33	97.27	95.91	93.40	94.64
	unseen	60.16	82.99	76.11	66.05	70.72

Table 4: Comparison of SES+SVM to Freeling on the lemmatization task for Catalan

the fact that we have been able to train and evaluate it on eight typologically diverse languages in a short period of time. Our system is context-sensitive: it incorporates features of context words surrounding the target word form to combine lemmatization with disambiguation.

Though certainly useful, lemmatization without accompanying morphological analysis is often insufficient. An interesting topic for future research is the question of how to best integrate our approach with more detailed inflectional analysis. In a language with moderately rich morphology such as Spanish, inflectional information can be encoded in part-of-speech tags and machine-learning techniques can be used to induce them, either independently or in conjunction with lemmatization. For inflectionally denser languages such as Polish this may turn out to be more problematic. An interesting idea to explore would be to use a lexicon-based analyzer and combine it with a data-driven approach which would be used for word-forms unseen in the dictionary.

As evident from the Dutch results, our approach is inadequate when the assumption of suffixal inflection does not hold. It would be interesting to investigate whether, and how, SES-based lemmatization can be modified to deal with those cases.

References

Abeillé, Anne, Lionel Clément, and François Toussenel. 2003. Building a treebank for French. In A. Abeillé, editor, *Treebanks: Building and Using Parsed Corpora*. Kluwer Academic Publishers, Dordrecht, pages 165–188.

Afonso, Susana, Eckhard Bick, Renato Haber, and Diana Santos. 2002. "Floresta sintá(c)tica": a treebank for Portuguese. In *Proceedings of LREC 2002*, pages 1698–1703, Las Palmas de Gran Canaria.

Aho, Alfred V., Daniel S. Hirschberg, and Jeffrey D. Ullman. 1976. Bounds on the complexity of the longest common subsequence problem. *J. ACM*, 23(1):1–12.

Bień, Janusz S. and Marcin Woliński. 2003. Wzbogacony korpus słownika frekwencyjnego polszczyzny współczesnej. In Jadwiga Linde-Usiekniewicz, editor, *Prace lingwistyczne dedykowane prof. Jadwidze Sambor*. Wydział Polonistyki, Uniwersytet Warszawski, Warsaw, pages 6–10.

Boser, Bernhard E., Isabelle M. Guyon, and Vladimir N. Vapnik. 1992. A training algorithm for optimal margin classifiers. In *COLT '92: Proceedings of the fifth annual workshop on computational learning theory*, pages 144–152, New York, NY, USA. ACM Press.

Brants, Sabine, Stefanie Dipper, Silvia Hansen, Wolfgang Lezius, and George Smith. 2002. The TIGER treebank. In *Proceedings of the Workshop on Treebanks and Linguistic Theories*, Sozopol.

Carreras, Xavier, Isaac Chao, Lluís Padró, and Muntsa Padró. 2004. Freeling: An open-source suite of language analyzers. In *Proceedings of the 4th LREC Conference*, Lisboa.

Chang, Chih-Chung and Chih-Jen Lin. 2001. LIBSVM: a library for Support Vector Machines (version 2.31).

Civit, Monsterrat, Núria Bufí, and Pilar Valverde. 2004. Building Cat3LB: a treebank for Catalan. In *Proceedings of the SALTMIL Workshop at LREC 2004*, pages 48–51, Lisboa.

Civit, Montserrat. and Ma Antònia Martí. 2004. Building Cast3LB: A Spanish treebank. *Research on Language and Computation*, 2(4):549–574, December.

Erjavec, Tomaž and Sašo Džeroski. 2004. Machine learning of morphosyntactic structure: Lemmatizing unknown Slovene words. *Applied Artificial Intelligence*, 18:17–41, January.

Hirschberg, Daniel S. 1977. Algorithms for the longest common subsequence problem. *J. ACM*, 24(4):664–675, October.

Kurohashi, Sadao and Makoto Nagao. 2003. Building a Japanese parsed corpus while improving the parsing system. In A. Abeillé, editor, *Treebanks: Building and Using Parsed Corpora*. Kluwer Academic Publishers, Dordrecht.

Myers, Eugene W. 1986. An O(ND) difference algorithm and its variations. *Algorithmica*, 1(1):251–266, March.

Stroppa, Nicolas and François Yvon. 2005. An analogical learner for morphological analysis. In *Proceedings of the Ninth Conference on Computational Natural Language Learning (CoNLL-2005)*, pages 120–127, Ann Arbor, Michigan, June. Association for Computational Linguistics.

van den Bosch, Antal and Walter Daelemans. 1999. Memory-based morphological analysis. In *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*, pages 285–292, Morristown, New Jersey. Association for Computational Linguistics.

van der Beek, Leonoor, Gosse Bouma, Robert Malouf, and Gertjan van Noord. 2002. The Alpino dependency treebank. In *Computational Linguistics in the Netherlands CLIN 2001*. Rodopi.

Vapnik, Vladimir N. 1998. *Statistical Learning Theory*. Wiley-Interscience, New York.